Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«СЕВЕРО-КАВКАЗСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт цифрового развития

Кафедра инфокоммуникаций

**ОТЧЕТ**

**ПО ЛАБОРАТОРНОЙ РАБОТЕ №2.6**

**дисциплины «Программирование на Python»**

**Вариант\_\_\_**

|  |  |  |  |
| --- | --- | --- | --- |
|  | | Выполнил:  Иващенко Олег Андреевич  2 курс, группа ИВТ-б-о-22-1,  09.03.02 «Информационные и вычислительные машины», направленность (профиль) «Программное обеспечение средств вычислительной техники и автоматизированных систем»  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | | Руководитель практики:  Воронкин Роман Александрович, доцент кафедры инфокоммуникаций  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) | |
|  | |  | |

Отчет защищен с оценкой \_\_\_\_\_\_\_\_\_\_\_ Дата защиты\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ставрополь, 2023 г.

**Тема**: «Работа со словарями в языке Python»

**Цель**: Приобретение навыков по работе со словарями при написании программ с помощью языка программирования Python версии 3.x.

Порядок выполнения работы

Таблица 1 – Код программы example.py

|  |
| --- |
| #!/usr/bin/env python3  # -\*- coding: utf-8 -\*-  import sys  from datetime import date  if \_\_name\_\_ == '\_\_main\_\_':  # Список работников.  workers = []  # Организовать бесконечный цикл запроса команд.  while True:  # Запросить команду из терминала.  command = input(">>> ").lower()  # Выполнить действие в соответствие с командой.  if command == 'exit':  break  elif command == 'add':  # Запросить данные о работнике.  name = input("Фамилия и инициалы? ")  post = input("Должность? ")  year = int(input("Год поступления? "))  # Создать словарь.  worker = {  'name': name,  'post': post,  'year': year,  }  # Добавить словарь в список.  workers.append(worker)  # Отсортировать список в случае необходимости.  if len(workers) > 1:  workers.sort(key=lambda item: item.get('name', ''))  elif command == 'list':  # Заголовок таблицы.  line = '+-{}-+-{}-+-{}-+-{}-+'.format(  '-' \* 4,  '-' \* 30,  '-' \* 20,  '-' \* 8  )  print(line)  print(  '| {:^4} | {:^30} | {:^20} | {:^8} |'.format(  "№",  "Ф.И.О."  "Должность",  "Год"  )  )  print(line)  # Вывести данные о всех сотрудниках.  for idx, worker in enumerate(workers, 1):  print(  '| {:>4} | {:<30} | {:<20} | {:>8} |'.format(  idx,  worker.get('name', ''),  worker.get('post', ''),  worker.get('year', 0)  )  )  print(line)  elif command.startswith('select '):  # Получить текущую дату.  today = date.today()  # Разбить команду на части для выделения номера года.  parts = command.split(' ', maxsplit=1)  # Получить требуемый стаж.  period = int(parts[1])  # Инициализировать счетчик.  count = 0  # Проверить сведения работников из списка.  for worker in workers:  if today.year - worker.get('year', today.year) >= period:  count += 1  print(  '{:>4}: {}'.format(count, worker.get('name', ''))  )  # Если счетчик равен 0, то работники не найдены.  if count == 0:  print("Работники с заданным стажем не найдены.")  elif command == 'help':  # Вывести справку о работе с программой.  print("Список команд:\n")  print("add - добавить работника;")  print("list - вывести список работников;")  print("select <стаж> - запросить работников со стажем;")  print("help - отобразить справку;")  print("exit - завершить работу с программой.")  else:  print(f"Неизвестная команда {command}", file=sys.stderr) |

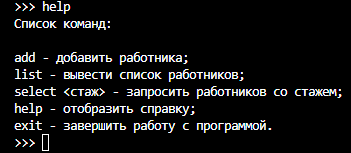


Рисунок 1.1 – Вывод команды help программы example.py

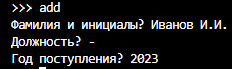


Рисунок 1.2 – Вывод команды add и добавление нового сотрудника в программе example.py

Задание 1. Решите задачу: создайте словарь, связав его с переменной school, и наполните данными, которые бы отражали количество учащихся в разны классах (1а, 1б, 2б, 6а, 7в и т.п.). Внесите изменения в словарь согласно следующему: а) в одном из классов изменилось количество учащихся, б) в школе появился новый класс, с) в школе был расформирован (удалён) другой класс. Вычислите общее количество учащихся в школе.

Таблица 2 – Код программы individual\_1:

|  |
| --- |
| #!/usr/bin/env python3  # -\*- coding: utf-8 -\*-  if \_\_name\_\_ == "\_\_main\_\_":  school = []  while True:  cmd = input(">>> ")  if cmd == "exit":  print("Завершение работы программы...")  break  elif cmd == "help":  print("add - добавление нового класса")  print("list - вывод список текущих классов")  print("remove - удаление класса")  print("change - изменение существующего класса")  print("getsum - вывод общего количества учащихся")  print("exit - выход из программы")  elif cmd == "add":  class\_name = input("Название класса: ")  class\_students = input("Количество учеников: ")  new\_class = {  'name': class\_name,  'students': class\_students  }  school.append(new\_class)  elif cmd == "list":  if len(school) > 0:  for object\_class in school:  print(f"Класс {object\_class['name']}. "  f"Количество учеников: {object\_class['students']}"  )  else:  print("Нет созданных классов")  elif cmd == "remove":  remove\_class\_name = input("Введите удаляемый класс: ")  for object\_class in school:  removed = False  if object\_class['name'] == remove\_class\_name:  school.remove(object\_class)  removed = True  break  if removed == False:  print("Класс не был найден")  else:  print("Класс успешно удалён")    elif cmd == "change":  changed\_class = input("Введите название класса для изменения: ")  changed = False  for object\_class in school:  if object\_class['name'] == changed\_class:  object\_class['name'] = input("Введите новое имя класса: ")  object\_class['students'] = input("Введите новое "  "количество учеников: ")  changed = True  break    if changed == False:  print("Класс не найден")  else:  print("Класс изменён")  elif cmd == "getsum":  class\_sum = 0  for object\_class in school:  class\_sum += int(object\_class['students'])  print(f"Общее количество учащихся: {class\_sum}")  else:  print("Введённая команда не существует. Введите help") |

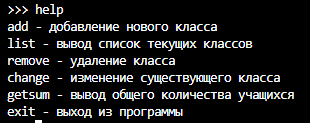


Рисунок 2.1 – Вывод команды help программы individual\_1.py

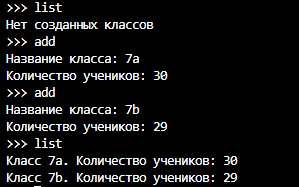


Рисунок 2.2 – Вывод всех классов и добавление новых в программе individual\_1.py

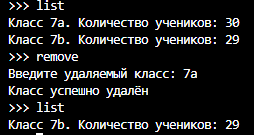


Рисунок 2.3 – Удаление классов в программе individual\_1.py

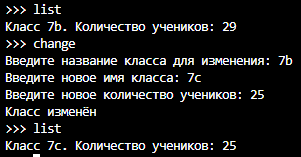


Рисунок 2.4 – Изменение существующего класса в программе individual\_1.py

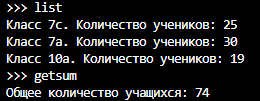


Рисунок 2.5 – Вывод общего количества учащихся в программе individual\_1.py

Задание 2. Создайте словарь, где ключами являются числа, а значениями – строки. Примените к нему метод items(), с помощью полученного объекта dict\_items создайте новый словарь, «обратный» исходному, т.е. ключами являются строки, а значениями – числа.

Таблица 3 – Код программы individual\_2.py

|  |
| --- |
| #!/usr/bin/env python3  # -\*- coding: utf-8 -\*-  if \_\_name\_\_ == "\_\_main\_\_":  first\_dict = {3: 'tree',  2: 'two',  1: 'one'}  second\_dict = dict(map(reversed, first\_dict.items()))  print(first\_dict.items())  print(second\_dict.items()) |



Рисунок 3 – Вывод программы individual\_3.py

Индивидуальное задание. Использовать словарь, содержащий следующие ключи: фамилия, имя, номер телефона, дата рождения (список из трёх чисел). Написать программу, выполняющую следующие действия: ввод с клавиатуры данных в список, состоящий из словарей заданной структуры; записи должны быть упорядочены по трём первым цифрам номера телефона; вывод на экран информации о человеке, чья фамилия введена с клавиатуры; если такого нет, выдать на дисплей соответствующее сообщение.

Таблица 4 – Код программы individual\_3.py

|  |
| --- |
| #!/usr/bin/env python3  # -\*- coding: utf-8 -\*-  if \_\_name\_\_ == "\_\_main\_\_":  member\_list = []  while True:  cmd = input(">>> ")  if cmd == "help":  print("add - добавление новых записей")  print("find - найти запись по фамилии")  elif cmd == "add":  surname = input("Введите фамилию: ")  name = input("Введите имя: ")  phone = input("Введите номер телефона: ")  date = tuple(map(int, input("Введите дату рождения: ").split('.')))    new\_member = {'surname': surname,  'name': name,  'phone': phone,  'date': date  }  member\_list.append(new\_member)  member\_list.sort(key=lambda item: item.get('phone')[:3])  elif cmd == "list":  for member in member\_list:  print(f"{member['surname']} {member['name']}, "  f"{member['phone']}, {member['date']}")    elif cmd == "find":  surname = input("Введите фамилию: ")  count = 0  for member in member\_list:  if member['surname'] == surname:  print(f"{member['surname']} {member['name']}, "  f"{member['phone']}, {member['date']}")  count += 1    if count == 0:  print("Записи не найдены")  elif cmd == "exit":  print("Завершение работы программы...")  break  else:  print(f"Команды {cmd} не существует") |

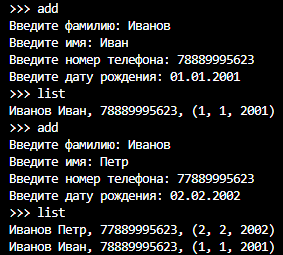


Рисунок 4.1 – Добавление новых записей и сортировка по номеру телефона в программе individual\_3.py

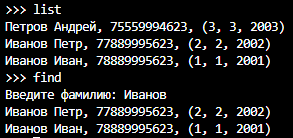


Рисунок 4.2 – Поиск записей по фамилии в программе individual\_3.py

Контрольные вопросы

1. Что такое словари в языке Python?

Словарь в Python – это изменяемая структура данных, представляющая собой набор пар ключ-значение. Ключи уникальны в пределах словаря, и они используются для доступа к соответствующим значением.

1. Может ли функция len() быть использована при работе со словарями?

Да, функция len() может быть использована для получения количества элементов (пар ключ-значение) в словаре.

1. Какие методы обхода словарей Вам известны?

Наиболее распространённые методы обхода словарей включают использование цикла for для перебора ключей и пар ключ-значение. Методы keys(), values() и items() также могут использоваться для получения ключей, значений или пар ключ-значение соответственно.

1. Какими способами можно получить значения из словаря по ключу?

Значения из словаря можно получить, используя квадратные скобки и указывая ключ, например:

value = new\_dict[‘ключ’]

1. Какими способами можно установить значение в словаре по ключу?

Значение в словаре можно установить или изменить, также используя квадратные скобки, например:

new\_dict[‘ключ’] = значение

1. Что такое словарь включений?

Словарь включений в Python – это компактный способ создания словаря в одной строке кода. Он позволяет создавать словари, используя цикл for для определения ключей и значений. Пример:

new\_dict = {k: k\*\*2 for k in range(1, 6)}

1. Самостоятельно изучите возможности функции zip() приведите примеры её использования.

Функция zip() в Python используется для объединения элементов из нескольких итерируемых объектов (например, два кортежа можно объединить в словарь с помощью метода zip([ключ], [значение])).

1. Самостоятельно изучите возможности модуля datetime. Каким функционалом по работе с датой и временем обладает этот модуль?

Модуль datetime предоставляет классы для работы с датой и временем. Он включает в себя классы datetime, date, time, timedelta и другие. С его помощью можно выполнять операции с датами, вычислять разницу между двумя датами, форматировать даты для вывода и многое другое.

**Выводы**: В процессе выполнения лабораторной работы были приобретены навыки по работе со списками при написании программ с помощью языка программирования Python версии 3.x, были написаны 4 программы: пример из лабораторной работы, 2 задания и индивидуальная задача.